Available online at www.sciencedirect.com

ScienceDirect

Procedia Technology 9 (2013) 442 – 450

Procedia Technology

CENTERIS 2013 - Conference on ENTERprise Information Systems / PRojMAN 2013 - International Conference on Project MANagement / HCIST 2013 - International Conference on Health and Social Care Information Systems and Technologies

Modeling ETL Data Quality Enforcement Tasks Using Relational Algebra Operators

Vasco Santos[a]*, Orlando Belo[b]

[a] *CIICESI, School of Management and Technology, Polytechnic of Porto, Rua do Curral, Casa do Curral, Apt. 205, 4610-156, Felgueiras, Portugal*

[b] *Algoritmi R&D Centre, Universidade do Minho, Portugal*

Abstract

Usually, a data warehouse is refreshed periodically with data gathered from disparate source systems. Nevertheless this data might not be fully accurate, probably containing serious data quality problems, such as uniqueness, misrepresentations, null values, multi-purpose fields, or inconsistent values, for one or more attributes. This is a major contribution to the falling expectations users have on data analyzed from data warehouses. Data quality enforcement is a complex time consuming task that parses data from source tables and corrects it, normalizes it and integrates it into a data warehouse for a better representation of real businesses. In this paper, we analyze some of the common tasks that are associated with data quality enforcement, representing and modeling them using Relational Algebra as specification tool.

Keywords: Data Warehouses; ETL modeling; Data Quality Enforcement; Relational Algebra.

* Vasco Santos. Tel.: +351-255-314002; fax: +351-255-314120.
E-mail address: vsantos@estgf.ipp.pt.

doi:10.1016/j.protcy.2013.12.049

1. Introduction

The primary goal of the development of a *Data Warehousing System* (DWS) is to provide the organization with a conformed, integrated and centralized repository [1] that in turn can help management, through data analysis, make decisions based on facts instead of intuition. Nevertheless, the development of such a project is very resource demanding, with a high level of complexity and with previously unknown setbacks that can undermine the success of the project. The quality of source data is one of them. Source systems, which tend to be OLTP systems supporting regular business activities, normally contain inaccurate data, unknown or null data and sometimes inconsistent data that are spread generally by operational systems' objects, constraints or rules. This imposes to include in data warehouses' populating systems – usually recognized simply by Extract-Transform-Load (ETL) processes - cleansing tasks in order to detect and filter (and sometimes recovering) all those data anomalies in the source's data before loading it into a *data warehouse* (DW).

Usually, these tasks are implemented and executed through the use of proprietary tools that analyze and transform the data accordingly to predefined business requirements. Although ETL processes have been subject of extensive research mainly in the conceptual and logical design [2-5] as well as all the aspects related to data quality that are one of the most important issues on the success of any populating process. So, cleansing tasks are undoubtedly important. They are mainly concerned with identifying problems in metadata and data [6-8], i.e., inconsistencies at attribute and row level, rather than defining a strategy to execute the procedures needed to deal with those problems. The focus of this paper is on modeling an approach, using Relational Algebra (RA), to deal with Data Quality Enforcement (DQE) procedures, since dealing with data conciliation was already focused on a previous work [9].

The paper is organized as follows. In section 2 we will study briefly data quality problems, its issues and the most frequent approaches to deal with them. Next, in section 3, we propose a formal specification to deal with the tasks needed to clean and conform data through the use of Relational Algebra. Finally, in the last section, we present some conclusions and describe some future work.

Table 1. Data Quality Problems [7]

Single-Source Problems		*Multi-Source Problems*	
Schema Level (Lack of integrity constraints, poor schema design)	*Instance Level* (Data entry errors)	*Schema Level* (Heterogeneous data models and schema designs)	*Instance Level* (Overlapping, contradicting and inconsistent data)
- Uniqueness	- Misspellings	- Naming conflicts	- Inconsistent aggregating
- Referential integrity	- Redundancy duplicates	- Structural conflicts	- Inconsistent timing
	- Contradictory values		

2. Dirty Data and Data Quality

A DW normally receives information from several sources, frequently of heterogeneous nature. These sources are responsible to support day-to-day business activities, storing and providing data to ensure common tasks in regular operational services. Nevertheless, these business activities not always provide such systems with accurate or even known data. It is common that during the normal use of *information systems* (IS) data might not be available for input or data might be inaccurately introduced in the system due to some not expectable event like a human error. The balance between allowing attributes to be null or forcing their introduction is also a compromise to be made in the development of any IS. However, it is quite important to

keep in memory that business cannot be stalled by overzealous requirements of an IS. However, when building a DW, several heterogeneous data sources are normally considered for integration and data can be presented in a lot of different formats, i.e., relational databases, structured or semi-structured files, or even free-form files. When analyzing source data, the more permissive the rules are the more difficult it is to validate the data that was stored.

Analyzing the types of data quality problems present in source data [7], as classified in Table 1, it is notorious that cleaning tasks complexity is far greater when several sources are involved. When dealing with a single source, the majority of the problems with dirty data are a reflex of lack of constraints, integrity constraints, domain constraints or even referential integrity. If the schema is permissive, it is also normal to find at the instance level data misspelled, redundant or attributes that contradict each other, such as the attributes age attribute and date of birth, for instance. However, if multiple heterogeneous sources are used, the complexity of the problems associated with creating an integrated repository escalates to a higher degree of complexity. In this case we are dealing not only with data quality issues but also with data conciliation problems.

Although the problems associated with poor data quality are mostly identified, the tasks or procedures needed to deal with them are very diversified. In [10] a brief summary of actions is presented to solve data anomalies like the ones identified previously in Table 2. These tasks are normally executed during the transformation phase of an ETL process, which is mainly defined as a workflow process [11-14] and are normally executed in a dedicated environment (Fig. 1). However, uncommon DWS environments, such as grid environments, arose as an approach to deal with some of the ETL performance bottlenecks [15]. Nevertheless these environments are not compatible with the use of a Database Management System (DBMS) as support for data transformations. In order to take advantage of such environments a different grain must be used to support all data transformations needed, since the nodes present in the grid might be heterogeneous in terms of architecture and operating system.

Table 2. Anomalies Resolution Strategy [10]

N.	Description
1	Data Decomposition – obtain atomic values
2	Transformations:
2.1	Standards (uppercase, lowercase, acronyms and abbreviations)
2.2	Normalization (i.e. enforce business rules)
2.3	Corrections
3	Correct null values
4	Referential integrity enforcement
5	Data enrichment
6	Duplicate data resolution
7	Expert intervention
8	Enforce Data Domains
9	Enforce Mandatory Data Entry
10	Change Data Type
11	Change Multidimensional Model

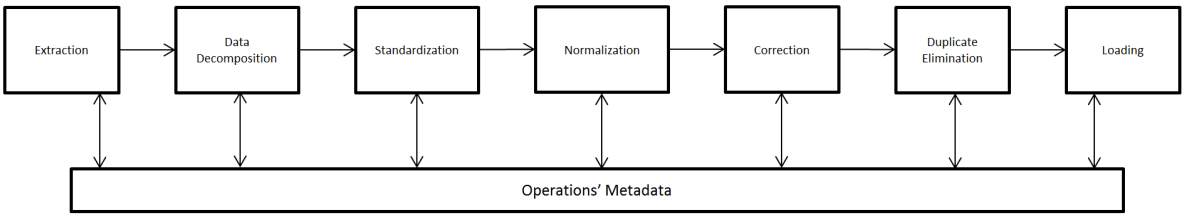

Fig. 1. Data Transformation and Cleaning Tasks [10]

Our choice fell upon the Relational Algebra language as support. In our opinion, it is an adequate formalism to specify such conceptual and logical tasks, having the advantage of not being platform or software dependent, as happens with SQL for instance. Based on that, our previous proposals presented a formalized approach of one of the most common ETL tasks, Slowly Changing Dimensions (SCD) [16], and also on the Data Conciliation phase [9], using Relational Algebra has support for specification. In the next section, we'll present a Relational Algebra modeling approach prepared specifically to deal with the data transformations and corrections needed to ensure the data quality desired in a centralized, integrated repository such as a DW.

3. Modeling DQE Tasks With Relational Algebra

Codd [17, 18] presented the Relational Model in the 70s. However, it stills supports today the most current databases, together with a data manipulation language called Relational Algebra. A great advantage this model has, and consequently its manipulation language, is that the data is stored in relations through the use of sets of tuples. The relational algebra presented inherits some of the properties of the mathematical Set theory, and has been, over the years, extended to support more data operations, like for instance aggregation [19, 20]. Later, Extended Relational Algebra was further developed to formally support duplicates, algebraic operations over attributes, and data manipulation like inserts, updates or deletes. Although the basis to support these operations is no longer Sets but the concept of Bags [21-23]. The primary difference between these two data structures is the ability to represent duplicates and the importance of data order present in Bags, which is not possible in Sets. With this in mind we now present in Relational Algebra equations and trees the most common operations identified previously that allow us to perform cleaning and conforming tasks to ensure a better level of data quality.

Let us first assume that the DW is based in the dimensional model [24] and therefore data extracted from sources will be cleaned, transformed and integrated into dimensions and fact tables. Tables, or relations, were defined in [17] and described in many subsequent articles and textbooks as a set n-tuples where each element of the tuple corresponding to a specific domain. For demonstration purposes we will use a general and common dimension structure presented in Eq. 1.

$$src\_dimdata_{S_x} = \langle BK_{S_x}, Att_1, \dots, Att_p \rangle \quad (1)$$

where BK_{Sx} is the business key of source S_x and $1 \leq x \leq n$, and $Att_1, \dots, Att_p$ are additional data attributes needed for the dimension.

3.1. Data Decomposition

Let us assume now that the attribute Att_p in Eq. 1 is an alphanumeric attribute that was used to store information that according to the data warehouse structure is now stored in several different attributes, for example we will define three new attributes, as defined in Eq. 2.

$$src\_dimdata\_d_{S_x} = \langle BK_{S_x}, Att_1, \dots, Att_p, Att_x, Att_y, Att_z \rangle \quad (2)$$

This is a common example of the use of data decomposition task. The contents of attribute Att_p should be decomposed into three new attributes Att_x, Att_y, Att_z. The Relational Algebra expression that will allow us to obtain that decomposition is presented in Eq. 3.

$$src\_dimdata\_d_{S_x} \leftarrow \varepsilon_{[Att_x = subs(Att_p,1), Att_y = subs(Att_p,2), Att_z = subs(Att_p,3)]}(src\_dimdata_{S_x}) \quad (3)$$

where the function *subs* is a user-defined function that extracts a set of characters from an attribute. The first argument is the attribute to be parsed and the second argument of the function defines which set of characters is intended (first, second, etc.). Generalizing, using the extend operator, ε [25], that creates new attributes based on data present in other attributes, either by algebraic operations or user-defined operations, it is possible to decompose the contents of any attribute.

3.2. Standardization

The standardization task has the goal to conform data that came from distinct sources in different formats but that has the same meaning. This happens frequently in cases that are associated with the use of upper or lower case characters in attributes and with the use, or misuse, of acronyms and abbreviations. Although recognized as standardization, there are a lot of different approaches to deal with the problems raised by these cases. In the case of upper and lower case characters, a system (or an user-defined function) will suffice in conforming the data. However dealing with acronyms and abbreviations requires the use of auxiliary tables to match and substitute the attribute to be treated. Unfortunately, it happens often to be necessary to do the standardization task by hand.

3.2.1. Upper and lower case characters

For this task, we assume that attribute Att_x, which was previously generated by the standardization operation, should be in upper case letters. Therefore, Eq. 4 presents the creation of a new attribute based on the Att_x attribute but with all letters in uppercase, and Eq. 5 removes the old Att_x attribute from table preserving the new one. Finally, Eq. 6 renames the new attribute to the old name restoring the original structure of the table.

$$Temp1 \leftarrow \varepsilon_{[Att_{x1} = upper(Att_x)]}(src\_dimdata\_d_{S_x}) \quad (4)$$

$$Temp2 \leftarrow \pi_{BK_{S_x}, Att_1, \dots, Att_p, Att_{x1}, Att_y, Att_z}(Temp1) \quad (5)$$

$$src\_dimdata\_u_{S_x} \leftarrow \rho_{Att_x / Att_{x1}}(Temp2) \quad (6)$$

With Relational Algebra trees (Fig. 2) it's quite simple to show how this set of operations works and understand clearly the sequence of operations followed. If there was a need to convert an attribute to a lower case then the approach to be follow would be similar to the presented in Fig. 2.

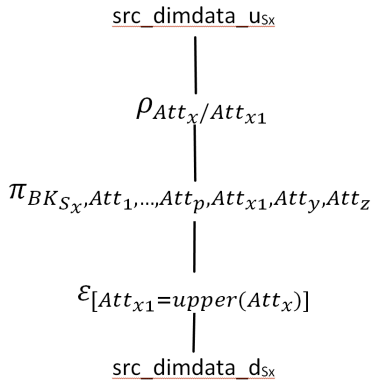

Fig. 2. Relational Algebra tree representation for upper case transformations

3.2.2. Acronyms and abbreviations

To conform an attribute that is supposed to store abbreviations/acronyms of some sort, we first need to create an auxiliary table integrating all possible values encountered in the source table and their correspondence in order to perform transformations required. In this particular case, and since there is a lookup in an auxiliary table, we also need to be aware that new representations of the abbreviations/acronyms might appear in the source table having no correspondence in the auxiliary table. In this case those tuples must be stored in a temporary table that will require an expert intervention in order to establish the correct correspondence in the auxiliary table and rerun the transformation for these tuples. In this case we assume that the attribute Att_y stores abbreviations but previous data analysis reports that there are different abbreviations for the same meaning.

Let us assume also that an auxiliary table was created to store all known correspondences (Eq. 7).

$$abb\_match = \langle Att_y, Att_{y1} \rangle \tag{7}$$

The task for conforming data is basically represented by the result of a join operation identifying the conformed abbreviation that will be stored in the DW (Fig. 3 (a)), and afterwards the identification of tuples that have attribute values with no match in the auxiliary table thus in need of an expert attention (Fig. 3 (b)). The proposed operations reflect the steps needed to conform abbreviations, but the approach to deal with acronyms is the same.

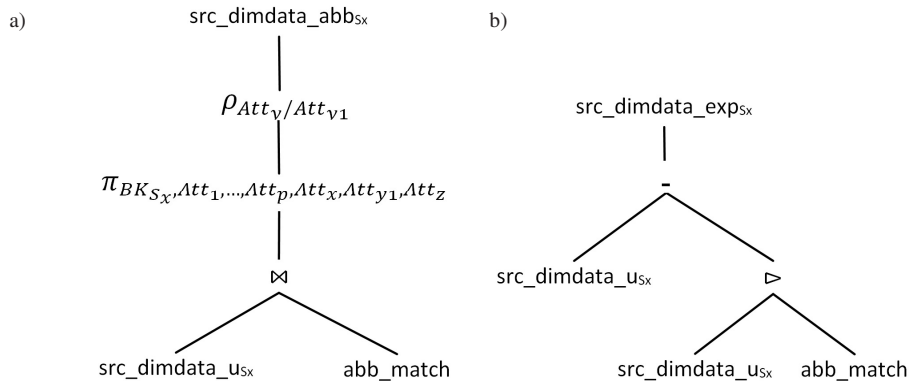

Fig. 3. (a) Conforming abbreviations and (b) identifying non correspondences

3.2.3. Normalization and Correction

The majority of the cleaning tasks needed to be done in the ETL process fall into this category, where data was introduced in the OLTP systems with errors. The common approach to deal with these errors is the same has presented in the previous task. First, we need to identify variations of the correct value, build an auxiliary table with all valid correspondences (or mappings) between dirty values and clean values, and then conforming the data as necessary. Data not treated should be stored in a quarantine data staging to be examined latter by an expert that will diagnose and correct (if possible) the problems, providing a correction patch that will be applied in future populating processes, i.e., insert all correspondences needed in the auxiliary table.

3.2.4. Duplicate Elimination

It is well known that identifying duplicates in source data is not an easy task. It is even more difficult to deal with these duplicates after detecting them. To avoid future problems, data in the source systems should be corrected in order to avoid future duplicates. However, in order to identify duplicates there must be an attribute that should be unique, but for some reason is not, that will allow us to determine these duplicates and deal with them.

Let us assume for our demonstration that the attribute Att_z stores information that should be unique in the dimension, i.e. the attribute Att_z, which was part of a larger attribute in the beginning of our demonstration identifying each tuple. One method to identify duplicates is to find if a given data instance is present in more than one tuple and then violating the uniqueness of the value stored in Att_z (Fig 4). If such records exist then they must be stored apart for expert supervision.

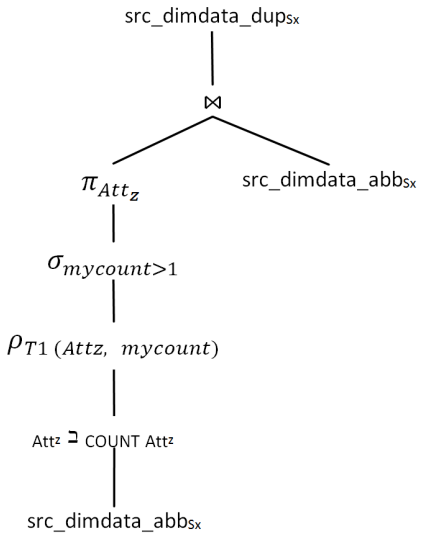

Fig. 4. Eliminating duplicates

After identifying the duplicates, they should be removed from the dimension table in order to proceed to the next task, probably data loading. All the duplicate detections will await for expert intervention in a specific quarantine data staging (Eq. 8).

$$src\_dimdata\_final_{S_x} \leftarrow src\_dimdata\_abb_{S_x} - src\_dimdata\_dup_{S_x} \quad (8)$$

4. Conclusions and Future Work

In the DWS community an ETL process is recognized as a very critical process in the success of a DWS, once it is responsible for gathering the information from sources and transforming it for future integration into the DW. The transformation task is not a simple attribute mapping, data type conversion or attribute renaming one; it contains undoubtedly very important tasks of cleaning and conforming data to provide the DW with correct data that faithfully reflects what actually happened in the operational systems. The DW must use clean and conformed data to better support management decision-making processes with effectiveness. However due to the nature of many information systems that are supporting business activities – usually conventional OLTP systems -, data being gathered might not always be accurate and correctly stored. Therefore, the cleansing tasks of the ETL process must try to identify the data problems and inconsistencies, performing transformations in order to guarantee minimum data quality and prepare data for loading into the DW. In this paper we presented a proposal for modeling those common cleansing tasks using Relational Algebra as support language. We think that it is ideal for uncommon DWS environments where a database management system normally does not exist. Modeling the approaches to deal with the cleansing tasks, present in common ETL processes, in a generalized and platform independent form will undoubtedly contribute to a higher degree of flexibility and usability. Relational Algebra operations can be implemented in any platform and system provided that the data is stored in a structured file as for instance XML. The advantage of this approach is the ability to use common technological infrastructure (ex. desktop computers) to execute the ETL process without the need of a DBMS. In a near future, we intend to model other standard ETL tasks using relational algebra following the intent to maximize the enterprises resources in a low cost DW

environment and provide a formal platform for ETL conceptual modeling covering all major tasks of a populating system of a DW.

References

[1] Inmon WH, *Building the Data Warehouse*, 4th ed., Wiley Publishing, Inc, 2005.
[2] Vassiliadis P, Simitsis A, Skiadopoulos S, Conceptual modeling for ETL processes, in: Proceedings of the 5th ACM international workshop on Data Warehousing and OLAP, ACM Press, McLean, Virginia, USA; 2002, p. 14-21.
[3] Vassiliadis P, Simitsis A, Skiadopoulos S, On the Logical Modeling of ETL Processes, in: Proceedings of the 14th International Conference on Advanced Information Systems Engineering, Springer-Verlag; 2002, p. 782-786.
[4] Trujillo J, Luján-Mora S, A UML Based Approach for Modeling ETL Processes in Data Warehouses, in: *Conceptual Modeling - ER 2003*, Springer -Verlag, Berlin Heidelberg; 2003, p. 307-320.
[5] Akkaoui ZE, Zimanyi E, Defining ETL worfklows using BPMN and BPEL, in: Proceedings of the ACM twelfth international workshop on Data warehousing and OLAP, ACM, Hong Kong, China; 2009, p. 41-48.
[6] Hernández MA, Stolfo SJ. Real-world Data is Dirty: Data Cleansing and The Merge/Purge Problem. *Data Mining and Knowledge Discovery* 1998; 2: 9-37.
[7] Rahm E, Do HH. Data Cleaning: Problems and Current Approaches. *IEEE Data Engineering Bulletin* 2000; 23: 3-13.
[8] Lee ML, Lu H, Ling TW, Ko YT, Cleansing Data for Mining and Warehousing, in: 10th International Conference on Database and Expert Systems Applications, Florence; 1999, p. 751-760.
[9] Santos V, Belo O, Modelling ETL Conciliation Tasks Using Relational Algebra Operators, in: International Conference on Applied Informatics and Communication (ICAIC 2012), Shenzhen, China; 2012.
[10] Costa AMPM, A Gestão da Qualidade dos Dados em Ambientes de Data Warehousing na Prossecução da Excelência da Informação, in: Departamento de Informática, Universidade do Minho, Braga; 2006, p. 220.
[11] Vassiliadis P, Simitsis A, Skiadopoulos S, Modeling ETL activities as graphs, in: L.V.S. Lakshmanan (Ed.) Design and Management of Data Warehouses 2002, Proceedings of the 4th Intl. Workshop DMDW'2002, CEUR-WS.org, Toronto, Canada; 2002, p. 52-61.
[12] Simitsis A, Vassiliadis P, Sellis T, Logical Optimization of ETL Workflows, in: Proceedings of the 4th Hellenic Data Management Symposium, Athens, Greece; 2005, p. 55-65.
[13] Vassiliadis P, Simitsis A, Terrovitis M, Skiadopoulos S, Blueprints and Measures for ETL Workflows, in; 2005, p. 385-400.
[14] Tziovara V, Vassiliadis P, Simitsis A, Deciding the physical implementation of ETL workflows, in: Proceedings of the ACM tenth international workshop on Data warehousing and OLAP, ACM, Lisbon, Portugal; 2007, p. 49-56.
[15] Santos V, Oliveira B, Silva R, Belo O. Configuring and Executing ETL Tasks on GRID Environments - Requirements and Specificities. *Procedia Technology* 2012; 1: 112--117.
[16] Santos V, Belo O, Using Relational Algebra on the Specification of Slowly Changing Dimensions - A First Step, in: 6ª Conferência Ibérica de Sistemas e Tecnologias de Informação, Chaves, Portugal; 2011.
[17] Codd EF. A relational model of data for large shared data banks. *Commununications of the ACM* 1970; 13: 377-387.
[18] Codd EF. Extending the database relational model to capture more meaning. *ACM Transactions on Database Systems* 1979; 4: 397-434.
[19] Klug A. Equivalence of Relational Algebra and Relational Calculus Query Languages Having Aggregate Functions. *J. ACM* 1982; 29: 699-717.
[20] Ozsoyoglu G, Ozsoyoglu ZM, Matos VM. Extending relational algebra and relational calculus with set-valued attributes and aggregate functions. *ACM Trans. Database Syst.* 1987; 12: 566-592.
[21] Albert J, Algebraic Properties of Bag Data Types, in: Proceedings of the 17th International Conference on Very Large Data Bases, Morgan Kaufmann Publishers Inc.; 1991, p. 211-219.
[22] Grefen PWPJ, de By RA, A multi-set extended relational algebra: a formal approach to a practical issue, in: Data Engineering, 1994. Proceedings.10th International Conference; 1994, p. 80-88.
[23] Libkin L, Wong L. Query Languages for Bags and Aggregate Functions. *Journal of Computer and System Sciences* 1997; 55: 241-272.
[24] Kimball R, Ross M, *The Data Warehouse Toolkit - The Complete Guide to Dimensional Modeling*, Second ed., John Wiley & Sons, 2002.
[25] Baralis E, Widom J, An Algebraic Approach to Rule Analysis in Expert Database Systems, in: Proceedings of the 20th International Conference on Very Large Data Bases, Morgan Kaufmann Publishers Inc.; 1994, p. 475 -- 486.